


REMARKS/ARGUMENTS

This amendment responds to the Office Action mailed on January 30, 2004. Claims 1-14 are now pending. Claims 1-7 have been amended and new claims 8-14 have been added to more completely cover the subject matter of the present invention.

Claims 1 - 7 were rejected as being identically disclosed in U.S. Patent No. 5,862,325 of Reed. In making this rejection the Examiner indicates that all of the limitations of claims 1-7 are found in Reed. The applicant respectfully disagrees and traverses the rejection on the basis of the amended claims.

The Reed patent is directed to ways for automatically updating information in an object-oriented programming system. When an attribute of an object is updated, the new value of the attribute overwrites the older version and the effective date of the new value is stored. Thus, it keeps track of the last update. This is similar to the disclosure at page 1 of the present application in the background section. However, in Reed there is no ability to review the original value of the attribute because it does not persist after an update.

In the present invention, as indicated by the amended claims, the original value of an attribute is stored "relatively persistently" in the data object. In particular, the old value persists in the data object along with its effective time, even when a new value and its effective time are stored. Thus the old value persists in the face of the new value. The attribute, however, does not need to be permanently persistent, i.e., if the system is designed to track

{W:\03343\000I048000\00174711.DOC  }

Appl. No. 09/747,504

Amdt. Dated January 23, 2004


Reply to Office Action of Sept. 25, 2003

performance over a year, it is only necessary that the attributes persist for that year. However, it must be at least relatively persistent, i.e., it must persist at least over the relevant time period or context.

Further, unlike some prior art techniques, e.g., as disclosed in the Suzuki et al. paper mention in the application at page 3 & 14, a new object is not created for each new value of the attribute at a new time. Rather, the time related data in the object is increased. Thus, only the time variant information is increased, while the static attributes do not increase in size. This is a most efficient way of storing all of the necessary data.

Also, unlike Reed, new attributes can be added to the object and tracked thereafter.

The difference between Reed and the present invention can be illustrated by an example. Suppose there is a company XYZ and it is coded as an object with attributes such as products, employees, and stock prices. Each attribute would be at least relatively persistently stored with its effective time. When a new value or update was received, it would also be stored with its effective time, but the original value would persist. In Reed, the original value and its effective time would be overwritten and would no longer be available. If it were decided to track the stock price, it would be easy to do with the present invention. In Reed's system, it would be impossible, because the stock price would be updated with a new value and the old one would be lost. The stock price over time would be available in the Reed system only if Reed knew at the time the database was created that it would be important to track stock

{W:\03343\000I048000\00174711.DOC  }

Appl. No. 09/747,504
Amdt. Dated January 23, 2004
Reply to Office Action of Sept. 25, 2003

price. Even then, some special code would be written to create a report on the stock price in Reed's system. With the present invention, it is merely a query of the database.

Since the Examiner has not supplied a reference which discloses persistent storage of time or context attributes, and this element is in all of the claims, the Examiner has not made a *prima facie* case of obviousness. Therefore, all of the claims should be allowed.

It is believed that the statement in the claims that the attribute or method is “at least relatively persistently stored” is not a narrowing amendment made for a purpose related to patentability. The original claims required the storage of current and changed values of the attributes and methods without any requirement that they be overwritten. The examined cited the Reed patent in which the values are overwritten. Thus, the present change to the claims was merely to make the original intent of the claims more clear.

It should be noted that “temporal” in the preamble of claims 4- 7 has been deleted. These claim are broadly to a context, which is not limited to time. Thus, the deleted language was unduly limiting.

New claims 8 and 9 were added to cover the situation where a new attribute is added after the database is started. See the specification at page 6. Claim 10 was added to cover the situation where the application is executed from a past time context. The is supported by the specification at pages 17-18. Claim 11 is directed to the disclosure at page 47 concerning view a system before and after error correction. Claim 12 is directed to execution of the system with an assumption of future time in order to make predictions as explained in the specification


{W:\03343\000I048000\00174711.DOC 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 }

Appl. No. 09/747,504
Amdt. Dated January 23, 2004
Reply to Office Action of Sept. 25, 2003

at page 8. Claims 13 and 14 are directed to the creation of temporal or other context base objects as development tool for creating temporal or context attributes and methods as subclasses of the tool. This is described at pages 8-9.

The Reed reference makes use of meta-data (a common technique) and a time-stamp (also a common technique) to provide flexible communication between two computer constructs separated by a network. It does not provide a general framework for incorporation of "context," of which temporality is one example, into programming languages, object oriented computing environments, data persistence or memory management. Nor does its exemplary embodiment anticipate the use of "context binding" by a programming language or environment to solve a wide array of business computing problems as described throughout the present application.

For the reasons cited above, Applicant believes that the rejection of claim 1-7 over Reed is not well grounded. Since the new claims 8-14 all depend directly or indirectly from claims 1-7, Applicant further states that all of the claims are not in condition for allowance.

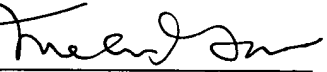
{W:\03343\000I048000\00174711.DOC  }

Appl. No. 09/747,504
Amdt. Dated January 23, 2004
Reply to Office Action of Sept. 25, 2003

A prompt and favorable action is earnestly solicited.


Respectfully submitted,

Dated: April 30, 2004

By: 

Melvin C. Garner
Reg. No. 26,272
Attorney for Applicants

Darby & Darby P.C.
Post Office Box 5257
New York, NY 10150-5257
212-527-7700

{W:\03343\000I048000\00174711.DOC  }

Appl. No. 09/747,504
Amdt. Dated January 23, 2004
Reply to Office Action of Sept. 25, 2003